CSBOT USER GUIDE

CsBot Version 2.6 (2013)

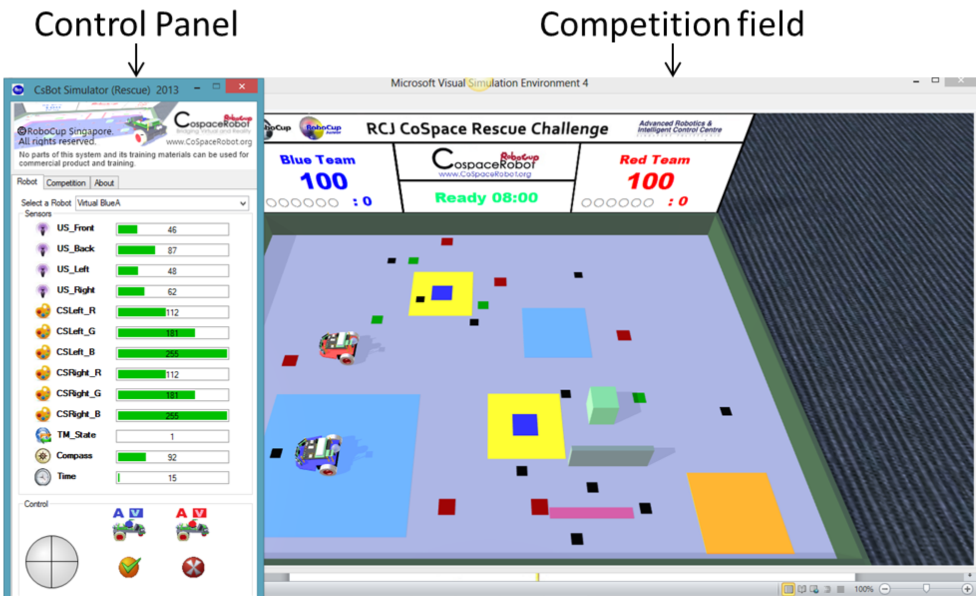
CONTENTS

1 Launching CsBot Software

To launch the CsBot Rescue application click on the CsBot icon: CsBot Rescue 2013

This will launch the software and will bring up a dialog box asking you to select wheather to use the secondary or primary platform. The split between primary and secondary teams is given in the rules on the cospace.org.uk website, please note that these catagories are not the same as UK school ages for primary and secondary.

After selecting your choice of platform a screen showing the competition field and the control pannel is launched. The control panel allows the user to manually control the robot and also view the live sensor values, it also allows programs to be created and the run.


Control Panel
Competition field
CsBot Simulator (Rescue) 2013
Microsoft Visual Simulation Environment 4
RCJ CoSpace Rescue Challenge
Blue Team 100
Red Team 100
Ready 08:00
Robot
Competition
About
Select a Robot Virtual BlueA
Sensors
Control

US_Front	46
US_Back	87
US_Left	48
US_Right	62
CSLeft_R	112
CSLeft_G	181
CSLeft_B	255
CSRight_R	112
CSRight_G	181
CSRight_B	255
TM_State	1
Compass	92
Time	15

2 CoSpace Virtual Robot

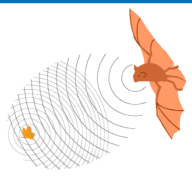
The virtual robots are shown in Figure 1 and are fitted with the following:

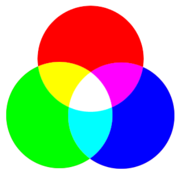
<u>Sensors</u>

- 4 ultrasound sensors that measure distance in centimetres fitted on the front, left, right and back of the robot. These can be used to detect walls and boundaries
- 2 colour sensors on the left and right of the robot that return red, green, blue values ranging between 0-255
- 1 Compass sensor at the top of the robot which measures the angle in degrees where 0° is facing the scoring board and the angle is measured anticlockwise

A small aside on how ultrasound and light sensors work….

<u>Ultrasound sensors</u> measure distance. This is done by transmitting an ultrasound wave and receives the echo when the ultrasound wave is reflected by an object. The time between the sending signal and the received echo is the distance to the object. This is the same method sonar methods that bats use to detect the distance of objects.





<u>Colour sensors</u> use photodiodes to detect the colour of objects beneath them. The colour detected is returned as 3 values, the red, green and blue values (rgb values) where each value is between 0-255. Any colour can be produced from a combination of R, G and B values.

Motors/Outputs

- 2 independently controlled motors at the back of the robot where setting the speed to 5 is the maximum speed forwards, -5 is the maximum speed backwards and 0 is stop.
- 1 LED to indicated when an object/deposition area has been found in accordance with the rules.

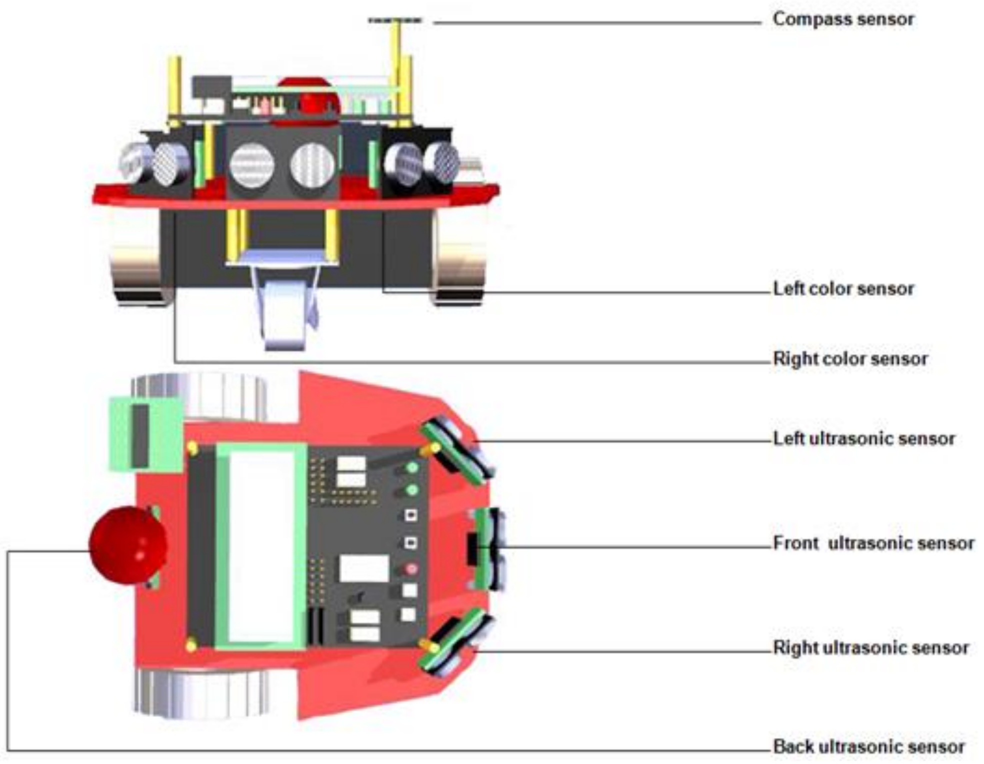
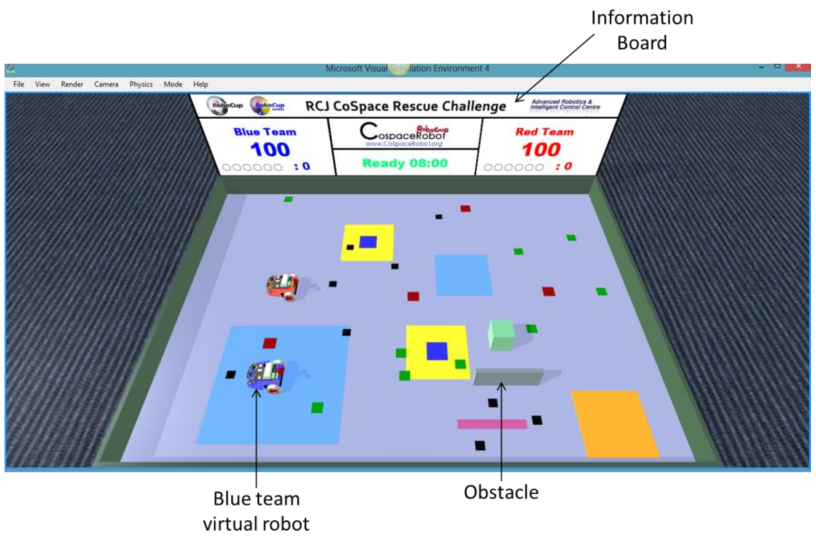


Figure 1 - CoSpace Virtual Robots

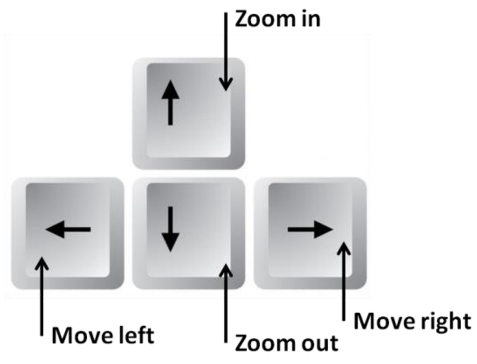
3 The Virtual Environment

Figure 2 shows the virtual environment used in CoSpace rescue, it consists of two virtual robots, some obstacles, and a set of different coloured objects. The information centre displays the team name, and the score



3.1 Navigating the virtual environment

Arrow keys can be used zoom and also to move around in the environment. You can also use the mouse to move around by clicking and dragging. The arrow keys work as follows:



Key	Action
q or page up	Move upwards
e or page down	Move downwards
'home' button	Return to initial position

If you hold the shift key whilst using on the keys provided you will move much faster.

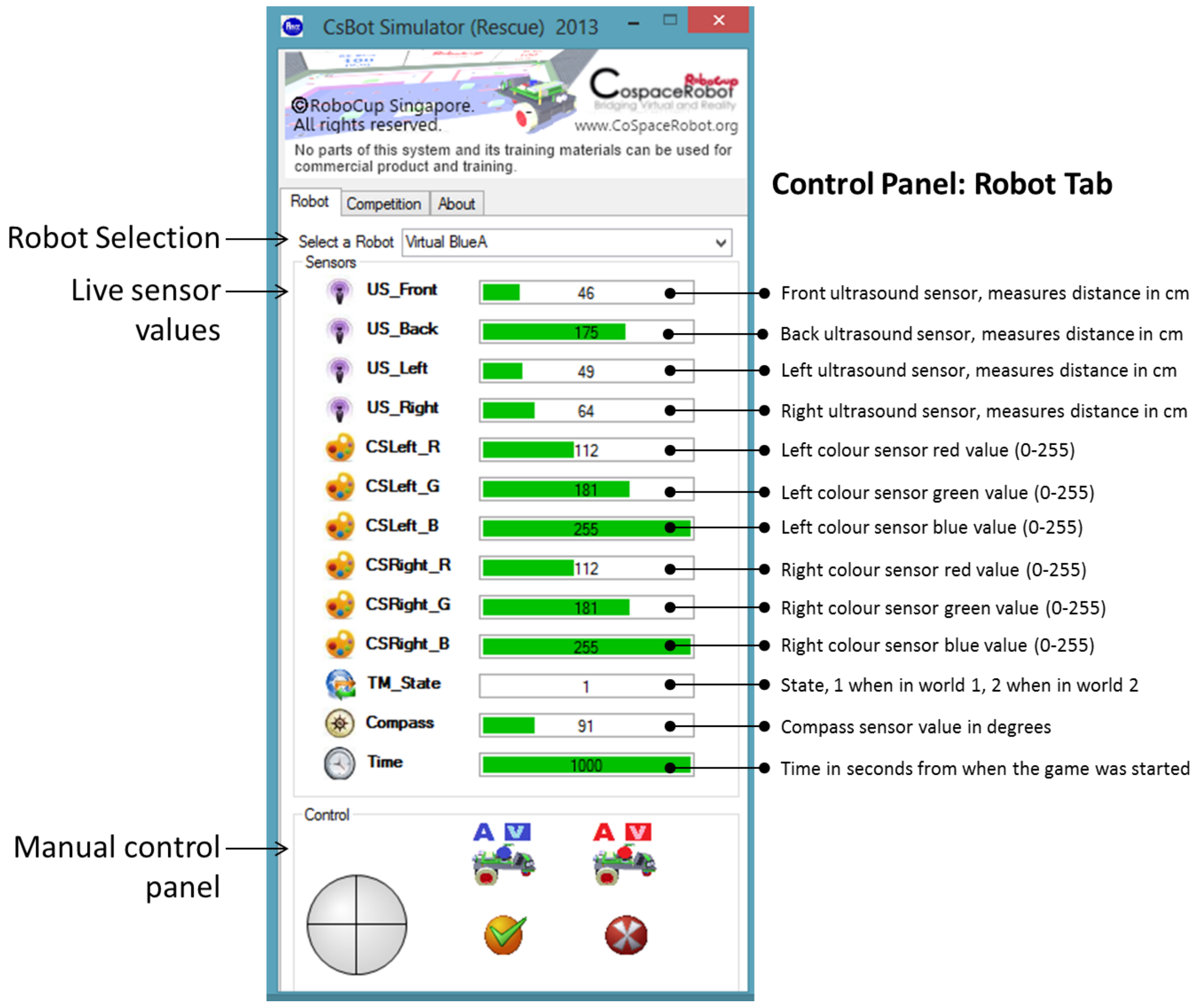
You can also navigate around by using the mouse and dragging the environment around.

4 Control panel

The control panel is launched with the software and is used to manually control the robot, to launch the AI programming window and also to run competitions. By clicking on the robot or competition tabs different options are given.

4.1 The Robot Tab

When the robot tab is selected you are able to view the real-time sensor values of the selected robot and control the real robot manually.



Sensor	Value	Description
US_Front	46	Front ultrasound sensor, measures distance in cm
US_Back	175	Back ultrasound sensor, measures distance in cm
US_Left	49	Left ultrasound sensor, measures distance in cm
US_Right	64	Right ultrasound sensor, measures distance in cm
CSLeft_R	112	Left colour sensor red value (0-255)
CSLeft_G	181	Left colour sensor green value (0-255)
CSLeft_B	255	Left colour sensor blue value (0-255)
CSRight_R	112	Right colour sensor red value (0-255)
CSRight_G	181	Right colour sensor green value (0-255)
CSRight_B	255	Right colour sensor blue value (0-255)
TM_State	1	State, 1 when in world 1, 2 when in world 2
Compass	91	Compass sensor value in degrees
Time	1000	Time in seconds from when the game was started

Robot selection – this allows the user to select which robots' sensor values to view.

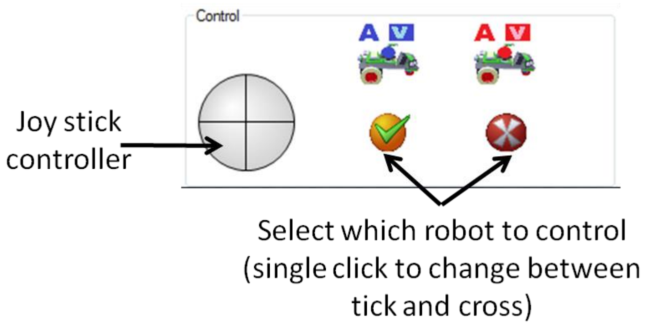
Real-time sensor feedback – this allows the user to view the sensors readings as the robot moves through the virtual environment. This is useful for calibrating the colour sensors.

Robot control – by moving the cross-hairs controller the direction and speed of the robot can be controlled manually, again this is useful in conjunction with the sensor values for calibration.

Even when the game is running it is possible to view this panel, making the sensor values useful for debugging.

4.1.1 Manual control of the robot(s)

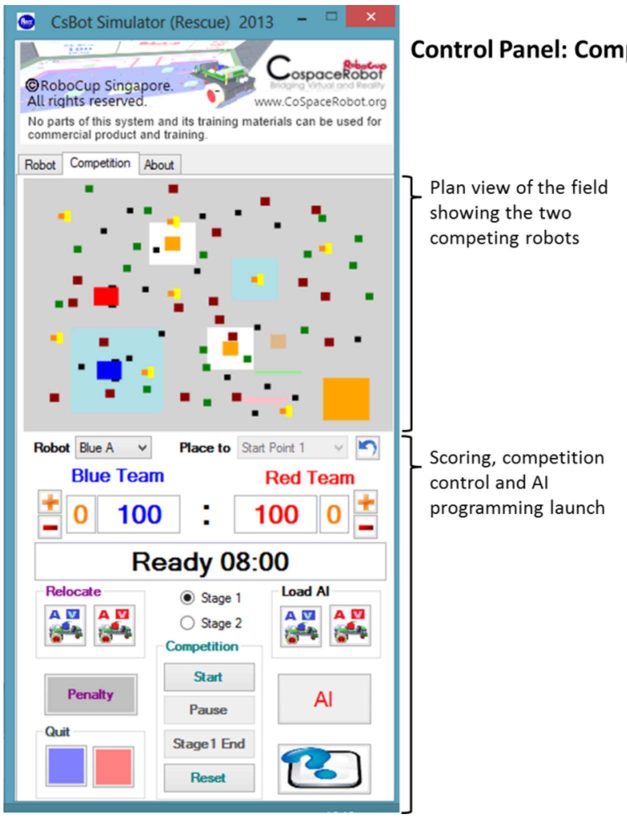
The joystick controller can be used to move the robot around the field. You can select which robot(s) to control by setting the


Control
Joy stick controller
Select which robot to control (single click to change between tick and cross)

control to a tick below the red or blue robot. You can select and move of the robots around at once.

4.2 The Competition Tab

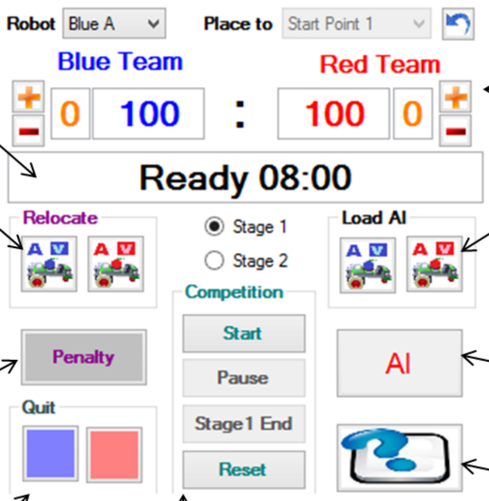
This tab of the control panel is used to load the programs written to run a competition and also to launch the AI development panel which is used to produce programs.


CsBot Simulator (Rescue) 2013
©RoboCup Singapore. All rights reserved.
CospaceRobot
www.CoSpaceRobot.org
No parts of this system and its training materials can be used for commercial product and training.
Robot
Competition
About
Robot
Blue A
Place to
Start Point 1
Blue Team
Red Team
0
100
:
100
0
Ready 08:00
Relocate
Stage 1
Stage 2
Load AI
Competition
Start
Pause
Stage1 End
Reset
Penalty
AI
Quit
Plan view of the field showing the two competing robots
Scoring, competition control and AI programming launch

Control Panel: Competition Tab

The bottom of this panel allows competitions to be started and allows the AI development panel to be launched:

Competition Panel



8 minute countdown timer

If one of the robots gets stuck in a loop or stuck on an obstacle you can click on the relocate button which will move the robot slightly

Click on this to award a penalty, a dialog box pops up asking to which robot to give the penalty to

If one of the teams want to quit on the game, they can click on these coloured buttons

Scoring for the two teams, automatically updates

To load a program into either the blue or red robot, click on the respectively coloured robot icon and select the program location when the dialog box opens

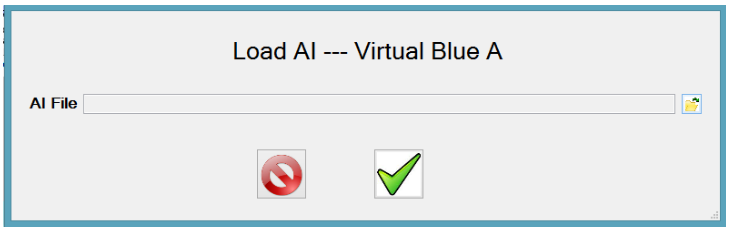
Launches the AI development panel for programming

Launches the help system

Please note parts of this are out of date

Start – starts a competition
Pause – pauses the competition
Stage 1 End – allows you to move immediately on to stage 2 when testing programs
Reset – resets the game, you will need to reload the programs

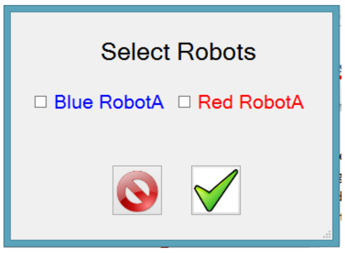
Loading a program



To do this click on the load AI button for the robot, this will launch the dialog window shown opposite. Click on the open file box and select the file to run, this will be a .dll file. Please note that you must do this EVERY time regardless of whether the file path is still shown on the dialog box. Once the file is selected, click on the green tick to load the program.

It is possible to load and run only one robot when testing. It is also possible to run two robots running the same program which can again be useful when testing.

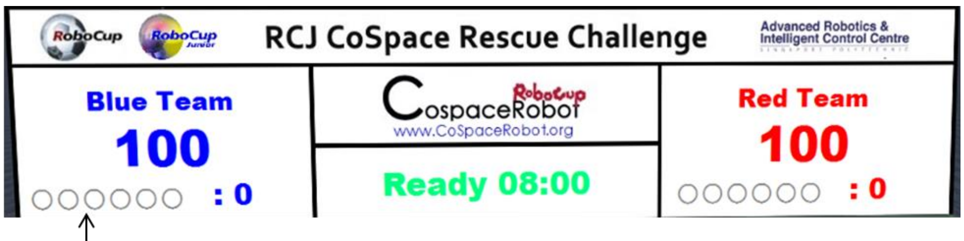
Penalties



When it is necessary to apply a penalty (as specified in the rules) click on the penalty box and the dialog box shown opposite with launch. Select the robot to which to apply the penalty is applied and then click the green tick. As stated in the rules, the robot will then be frozen for a period of time.

5 Scoreboard

There are two versions of the software which are used depending on version of the DDX (display) drivers that your computer has. If you have a new computer and having version 10.1 above you will have a version of software this score board. If you have a driver version 10 or then it does not support the graphics that are used to produce the score board so just use the competition tab on control panel for scoring.

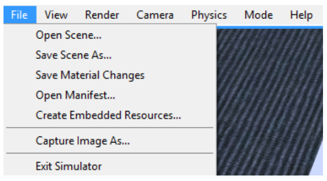


These six empty circles fill with the colour of objects that collected.

The score board will automatically update when objects are collected, deposited or lost. Remember that teams automatically start with 100 points.

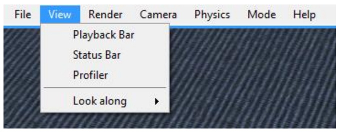
6 Menus in 3D environment

6.1 File Menu



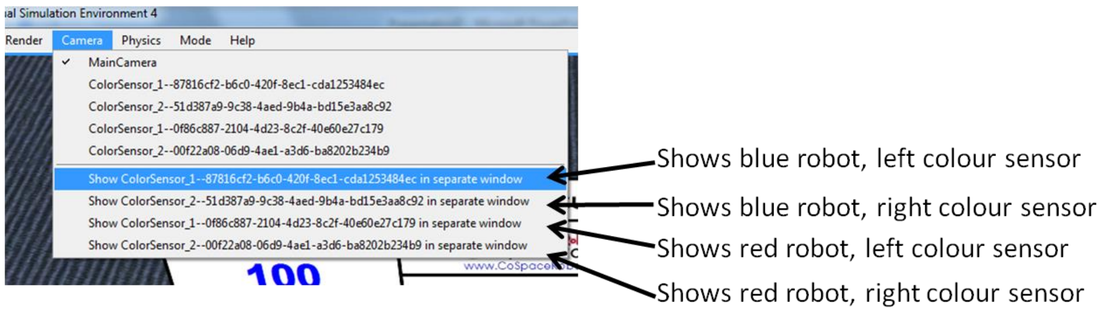
- Open Scene - Loads a scene.
- Save Scene As - Saves a scene. When you save a scene, the simulator saves the simulator state along with the state for every entity in the scene. It also saves a manifest which can be used to re-initialize the scene and any services associated with the entities.
- Save Material Changes - Saves changes made to materials in Edit mode.
- Open Manifest - Load a service manifest.
- Create Embedded Resources - Creates a single saved file containing all effects, textures, meshes, etc.
- Capture Image As - Save the current view of the simulation to a file.

6.2 View Menu



6.3 Camera Menu

This menu can be used to show what the colour sensor is seeing in a small separate window. Select one of the following option to launch the camera – this is very useful for calibration to ensure that that the sensor is correctly positioned over a green/red/black object.


al Simulation Environment 4
Render
Camera
Physics
Mode
Help
MainCamera
ColorSensor_1--87816cf2-b6c0-420f-8ec1-cda1253484ec
ColorSensor_2--51d387a9-9c38-4aed-9b4a-bd15e3aa8c92
ColorSensor_1--0f86c887-2104-4d23-8c2f-40e60e27c179
ColorSensor_2--00f22a08-06d9-4ae1-a3d6-ba8202b234b9
Show ColorSensor_1--87816cf2-b6c0-420f-8ec1-cda1253484ec in separate window
Show ColorSensor_2--51d387a9-9c38-4aed-9b4a-bd15e3aa8c92 in separate window
Show ColorSensor_1--0f86c887-2104-4d23-8c2f-40e60e27c179 in separate window
Show ColorSensor_2--00f22a08-06d9-4ae1-a3d6-ba8202b234b9 in separate window
Shows blue robot, left colour sensor
Shows blue robot, right colour sensor
Shows red robot, left colour sensor
Shows red robot, right colour sensor

This will launch a very small window – to make this bigger, left click carefully on the small blue edge once – this is fiddly and may take a few attempts.

7 Programming (the important bit!)

The programming used is event driven, and can be written using the graphical interface or using C# code. Alternatively a mix of the two can be used, creating a structure using the graphical interface to 'flesh' the program out.

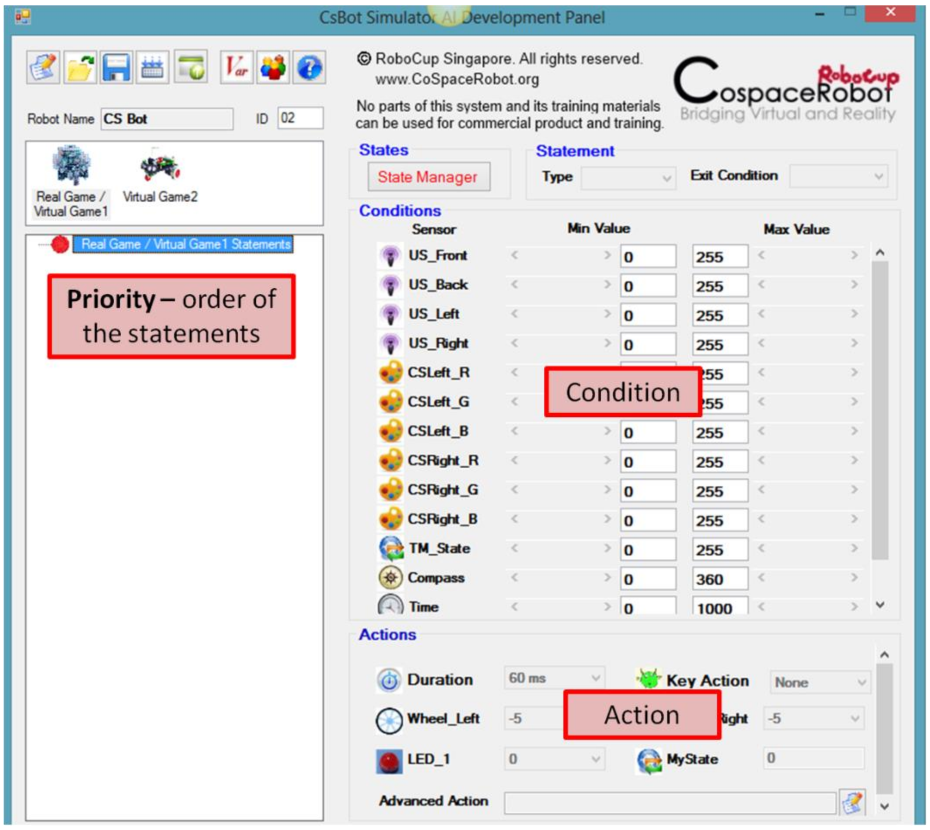
7.1 Event Driven programming – how it works

The programming used is event driven the flow of the program is determined by events. This is a different way of thinking about programming.

To program you create statements which have 3 properties associated with them:

1. Priority – set by the order of the statements
2. Condition
3. Action

These are set in the AI development panel:


CsBot Simulator AI Development Panel
© RoboCup Singapore. All rights reserved.
www.CoSpaceRobot.org
No parts of this system and its training materials can be used for commercial product and training.
CospaceRobot
RoboCup
Bridging Virtual and Reality
Real Game / Virtual Game1
Virtual Game2
Real Game / Virtual Game1 Statements
Priority – order of the statements
Condition
Action
States
State Manager
Statement
Type
Exit Condition
Conditions
Actions

Robot Name: CS Bot ID: 02

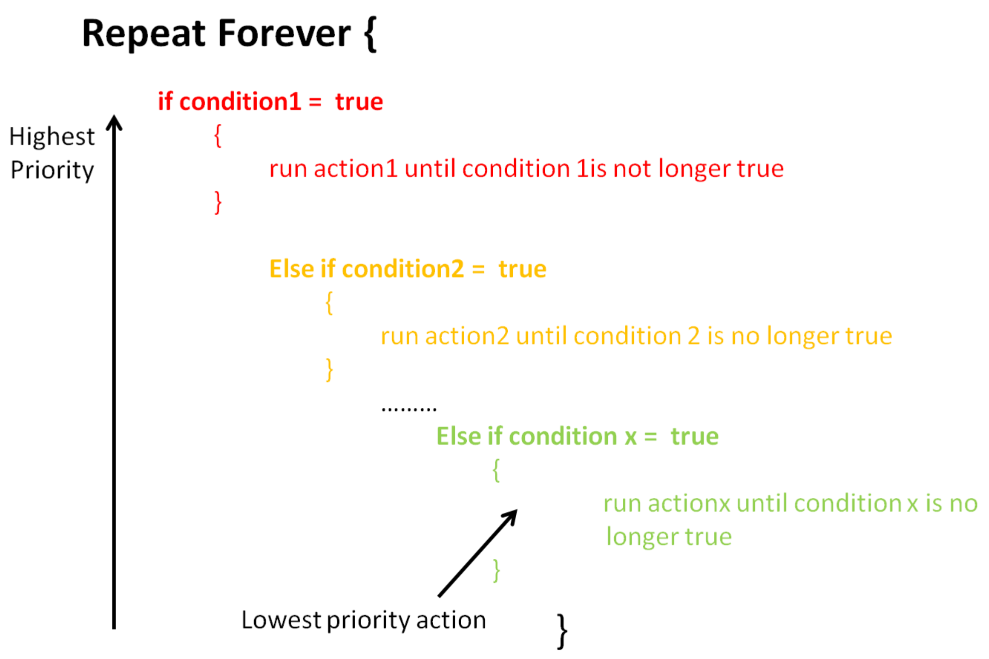
Sensor	Min Value	Max Value
US_Front	0	255
US_Back	0	255
US_Left	0	255
US_Right	0	255
CSLeft_R		55
CSLeft_G		55
CSLeft_B	0	255
CSRight_R	0	255
CSRight_G	0	255
CSRight_B	0	255
TM_State	0	255
Compass	0	360
Time	0	1000

Duration: 60 ms Key Action: None
Wheel_Left: -5 ight: -5
LED_1: 0 MyState: 0
Advanced Action

The program works by testing the condition of the highest priority statement, if this is true, it will run the action associate with this statement until the condition is no longer true.

If this highest priority statement is not true, the program will find the highest priority statement that is then true and run this action until it is no longer true. It will then return to the top, and start again, repeating this process forever.

This way the program works can be described using this pseudo code:



```
Repeat Forever {
    if condition1 =  true
        {
            run action1 until condition 1is not longer true
        }

            Else if condition2 =  true
                {
                    run action2 until condition 2 is no longer true
                }
                    .........
                        Else if condition x =  true
                            {
                                run actionx until condition x is no
                                longer true
                            }
                                                }
```

7.2 The AI Development Panel

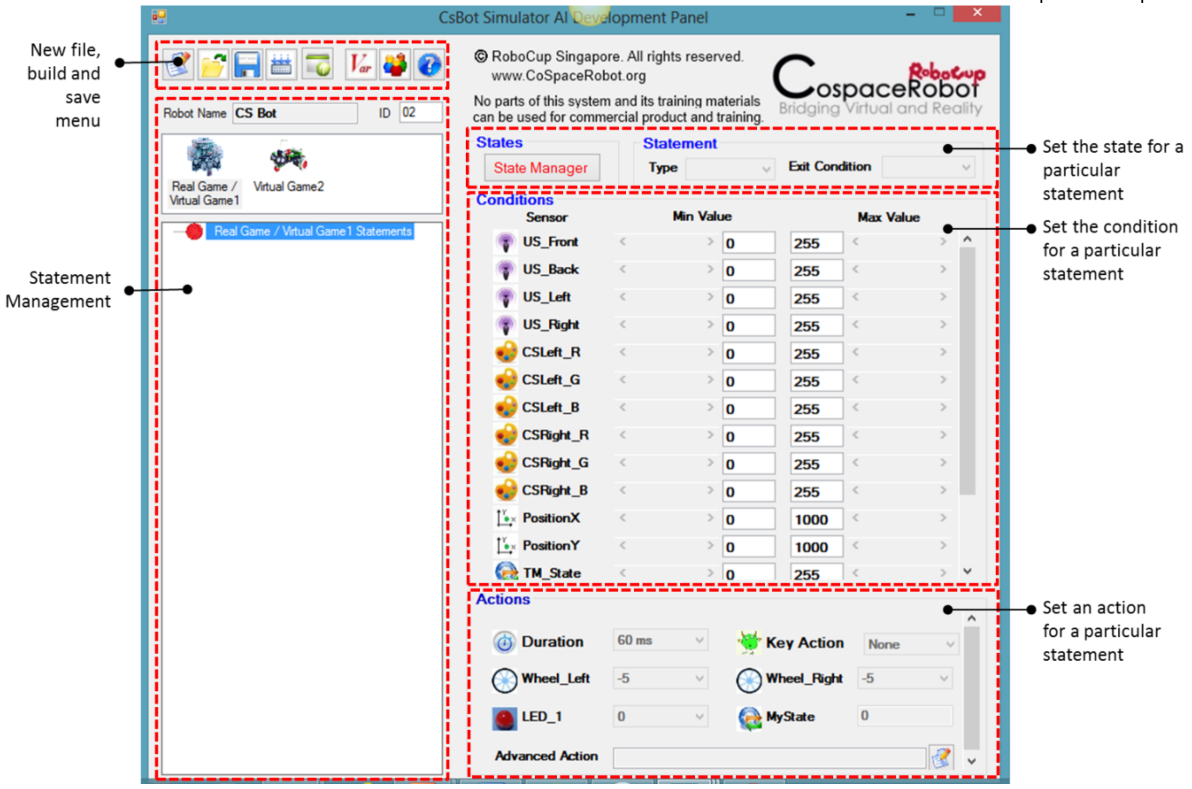
Clicking on the on the AI button launches the AI programming environment and will bring up the environment shown below. There are five main sections for this:

- File management section
- Statement management which allows you to add statements/bundles

For each statement you can then edit settings in:

- The state manager
- The condition manager

And you can set the action that should occur in the actions box.


New file, build and save menu
Statement Management
Set the state for a particular statement
Set the condition for a particular statement
Set an action for a particular statement
CsBot Simulator AI Development Panel
State Manager
Real Game / Virtual Game1
Virtual Game2
Real Game / Virtual Game1 Statements

© RoboCup Singapore. All rights reserved.
www.CoSpaceRobot.org
No parts of this system and its training materials can be used for commercial product and training.

CospaceRobot — Bridging Virtual and Reality

Robot Name: CS Bot   ID: 02

States — Statement: Type, Exit Condition

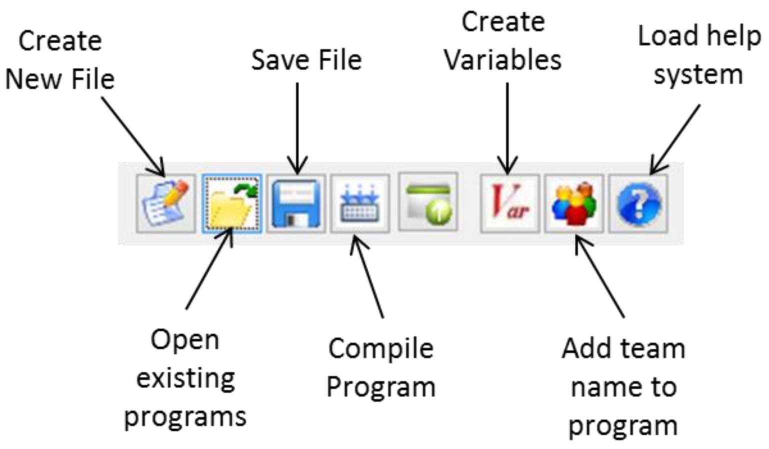
Conditions

Sensor	Min Value	Max Value
US_Front	0	255
US_Back	0	255
US_Left	0	255
US_Right	0	255
CSLeft_R	0	255
CSLeft_G	0	255
CSLeft_B	0	255
CSRight_R	0	255
CSRight_G	0	255
CSRight_B	0	255
PositionX	0	1000
PositionY	0	1000
TM_State	0	255

Actions

Duration	60 ms	Key Action	None
Wheel_Left	-5	Wheel_Right	-5
LED_1	0	MyState	0
Advanced Action			

File menu



7.3 Creating, loading, saving and building programs

The file menu can be used to create a new file, open existing programs, save programs and to compile the program.

7.3.1 Creating and initializing a new project

To create a new project click on the button. A dialog box will load asking you to provide a name for the project and the file path for the project.

7.3.2 Loading a project

To load an existing project click on the load button:

This will bring up a window which allows you to find and select a file. Project files have the extensions '.smp'. Once the file has been loaded, it can be edited.

7.3.3 Saving a project

To save a project file with the file extension of '.smp' click on the save button:

If a project has been loaded or previously saved it will automatically save to the same address, however it is also possible to svae with a different project name (a 'save as' feature) by changing the project name.

To load the program into the robot the file must be saved and built, not just saved, however if you just build and don't save the project file will not have the most up-to-date program.

7.3.4 Building a project

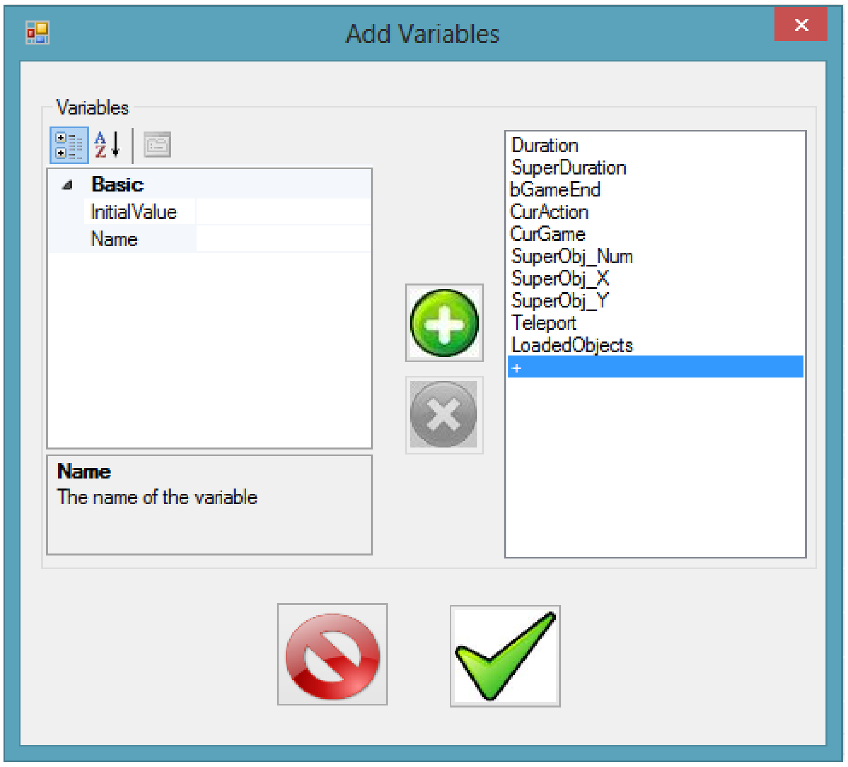
Before the program can be loaded onto the robot it must be 'built' this converts the .smp project file that has been created into a .dll file which can be loaded into the robot using the 'load AI function'. To build a project click on the build button:

Once the project has successful built (could take up to 5 seconds) there will be a pop-up saying that the program has built successfully.

7.4 Creating variables

All variables that are created are integer variables. There are some system variables that have already been created by the software.

To view the variables in your program click on the button the AI panel. This launches the 'add variables' panel as shown:



The variables shown initially are those created by the system.

To add new variables selected the + on the right hand side as shown. Enter a name and initial value into the left hand side panel and then select the green + icon to create the variable. Once the variable has been created you can exit the window by clicking on the green tick.

To delete a variable click on the variable and select the red cross which is below the green + button.

7.5 Adding a Team Name

The rules state that teams must add a team name to their program. This can be done by clicking on the team name button:

A dialog box will pop-up and ask you to enter a team name. Do this and then click on the accept button. You have now added a team name and this will be displayed on the score panel.

7.6 Statements

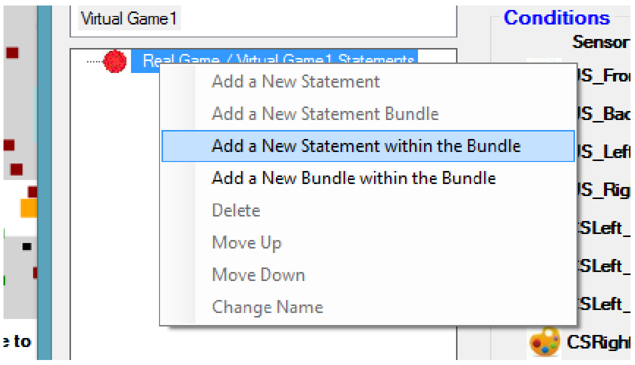
The CoSpace platform uses sequential programming technique. The compiler executes the program statements sequentially in a "top-down" manner. Therefore, the order of the statements in the program plays a very important role in deciding the priority

of statements with the same priority.In order to synchronise the real and virtual robots and maintain real-time data updating, the CoSpace platform automatically scans all sensors‟ readings in an interval of 60 ms. In other words, all the variables associated with sensors will be updated every 60ms.

7.6.1 Adding a new statement

Statement specifies action. A program is formed by one or more statements in sequence. Each statement will has a condition and an action associated with it. To add a new statement

1. Select the statement/subroutine to which you want the statement to be added to and click the right mouse button and select 'add a new statement within the bundle' if this is the first statement within the bundle. Once is a statement in the bundle you can select 'add a new statement'


Virtual Game1
Real Game / Virtual Game1 Statements
Add a New Statement
Add a New Statement Bundle
Add a New Statement within the Bundle
Add a New Bundle within the Bundle
Delete
Move Up
Move Down
Change Name
Conditions
Sensor
CSRight

2. Give the statement a meaningful name after which it will be added to the statement tree.

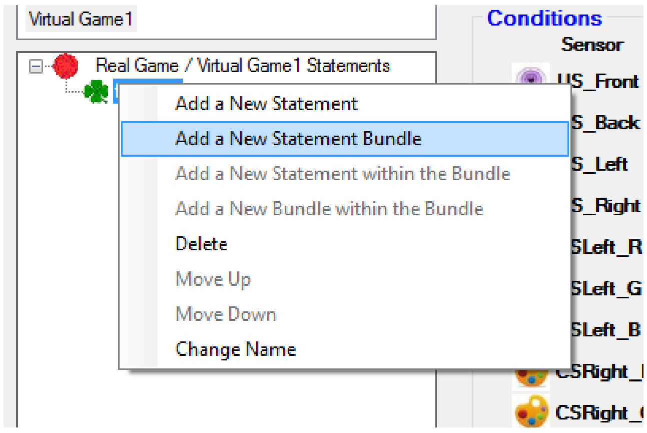
The statement type, conditions and actions associated with the new statement need to be specified. It will be illustrated in later section.

You can now continue to add more statements to the program.

7.6.2 Adding a new subroutine

You can add subroutines (also known as procedure or subprogram) which are referred to as 'bundles' of statements. They are a portion of code in the program which performs a specific task and can be used to decompose the program into smaller chunks. Using subroutines can allow your program to be less mess and more easy to debug and diagnose. To add a subroutine:

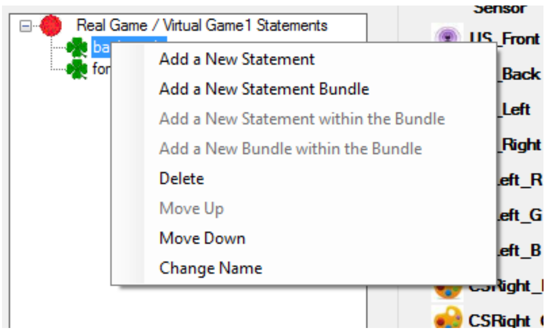
1. Select the previous statement/subroutine to which you want to place the subroutine after and right click this statement/subroutine. Select add 'add a new bundle within the bundle' or 'add a new bundle' as appropriate.


Virtual Game1
Real Game / Virtual Game1 Statements
Add a New Statement
Add a New Statement Bundle
Add a New Statement within the Bundle
Add a New Bundle within the Bundle
Delete
Move Up
Move Down
Change Name
Conditions
Sensor
US_Front

2. Assigning a meaningful name and then the subroutine is created and can be populated with more subroutines or statements.

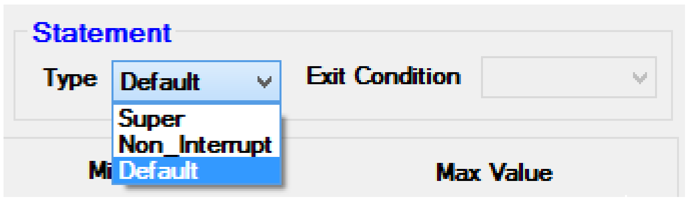
7.6.3 Managing statements and subroutines

The order of statements and bundles on the tree is very important as the higher statements are up on the tree the higher their priority. To delete, move up or move down a statement right click on the statement or subroutine and choose from the appropriate actions:


Real Game / Virtual Game1 Statements
Add a New Statement
Add a New Statement Bundle
Add a New Statement within the Bundle
Add a New Bundle within the Bundle
Delete
Move Up
Move Down
Change Name
Sensor
US_Front
Back
Left
Right
CSRight

7.6.4 Statement Type

When a statement is added, the **type of statement** needs to be set. The three type of statements that you can use for different requirements are a super statement, non-interrupt statement or a default statement as shown in figure below.



Default action – default statements have the lowest priority. Most statements in a project will be default statements.

Non-interrupt action – these statements have the same priority as default statements however when they are exected they will not be interrupted or terminated unless:

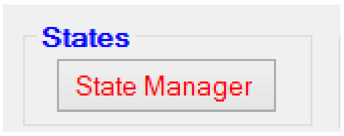
- The exit action condition is fulfilled
- The super action statement is executed

When the non-interrupt action is specified, it is necessary to define an exit condition for this action. Only when this specified exit condition is true will the statement be terminated. Projects can contain multiple non-interrupt action statements.

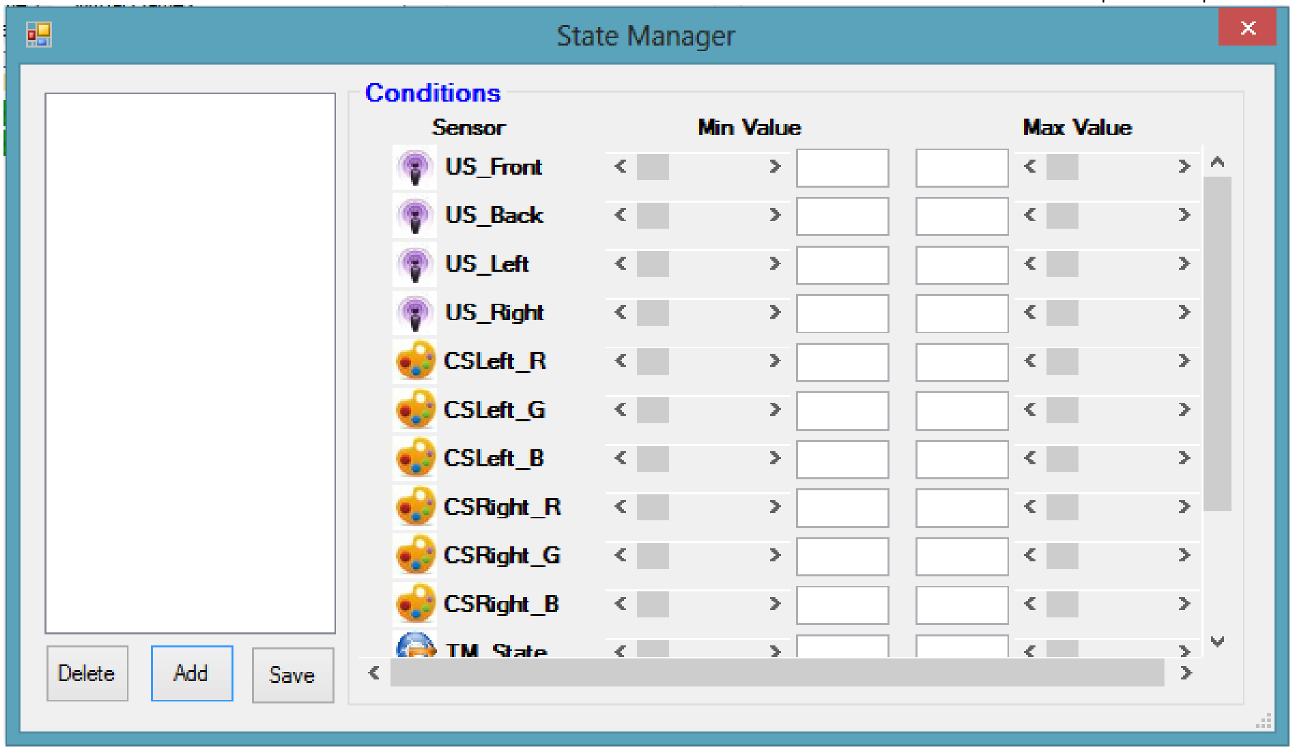
Super action – these statements have the highest priority. Once the condition for the statement is true it will be execute immediately, it will not wait for an action to complete, but will interrupt the action. A project can contain many super action statements.

7.6.5 Statement Manager

You need to set states for the exit condition of a non-interrupt statement. These can be set using the state manager. This can launched by selecting the state manager button in the AI environment.



Clicking on this button launches the state manager:



Sensor	Min Value	Max Value
US_Front		
US_Back		
US_Left		
US_Right		
CSLeft_R		
CSLeft_G		
CSLeft_B		
CSRight_R		
CSRight_G		
CSRight_B		
TM_State		

Adding a new state- double click on the 'add' button and enter a new state name. This must an an alphanumeric name (i.e. only contains letters and numbers.)

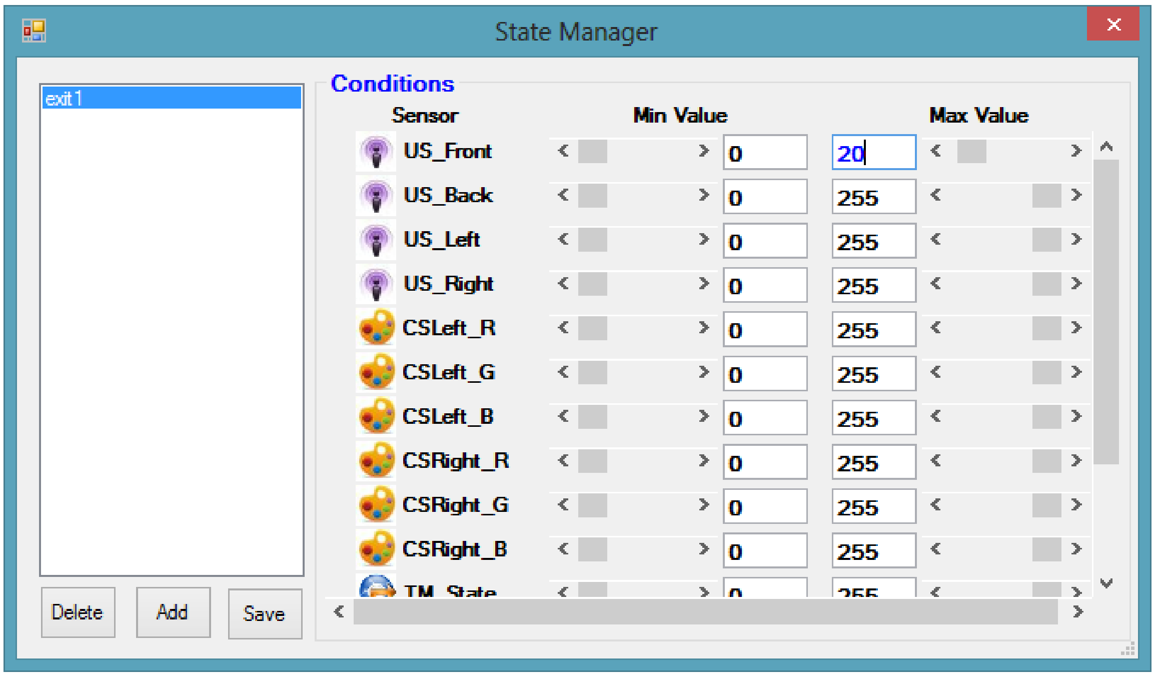
Deleting a state - select the state you want to delete and then click the delete button.

Saving a state - to save a state click on the save button, the state manager window will then be closed and the states will be saved.

7.6.6 Setting an exit condition for an non-interrupt statement

Launch the state manger. The condition for the state can then be added by using the sliders for the minimum and maximum values for the different sensor variables or by using the advanced condition option.

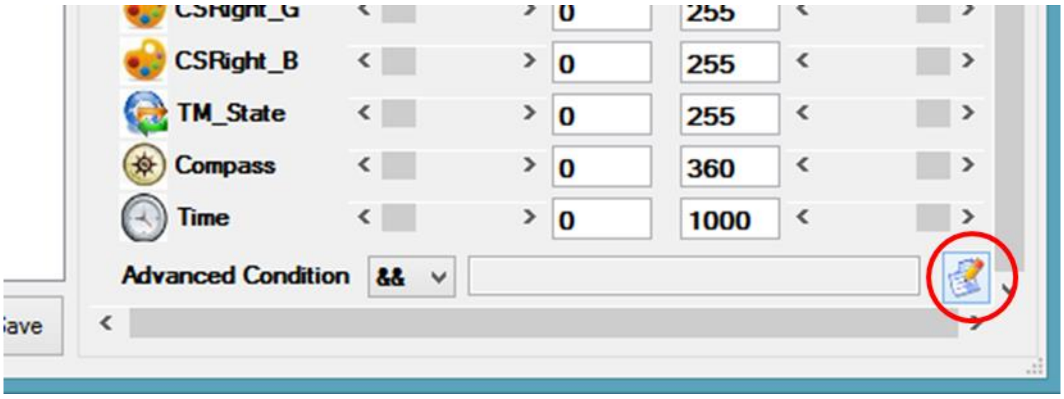
For example, to set the exit condition to be when the front ultrasound is less than 20 cm in a state called exit1 you would set the following:



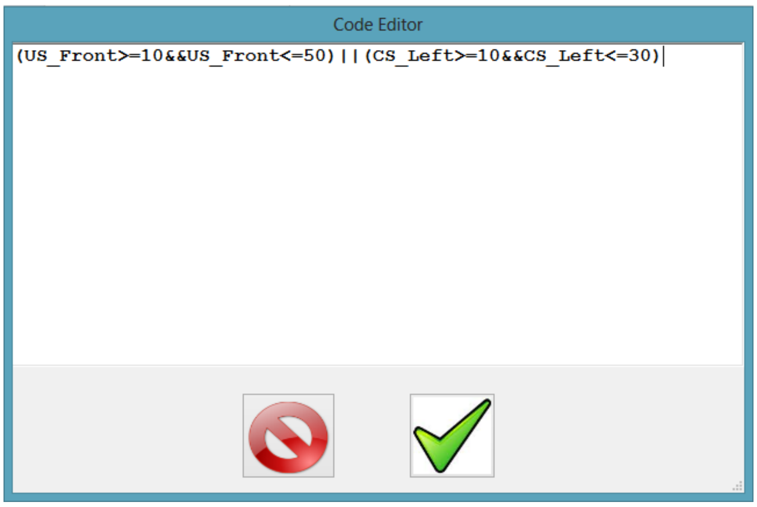
exit1

Sensor	Min Value	Max Value
US_Front	0	20
US_Back	0	255
US_Left	0	255
US_Right	0	255
CSLeft_R	0	255
CSLeft_G	0	255
CSLeft_B	0	255
CSRight_R	0	255
CSRight_G	0	255
CSRight_B	0	255
TM_State	0	255

To set more complicated conditions you can use to the advanced condition box. To do this, scroll to the bottom of the conditions section and select the 'write code' button:

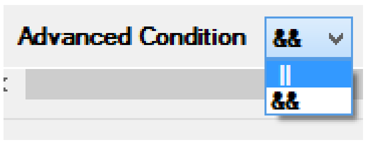


This will launch the code editor. You can then enter advanced conditions using C# code such as:

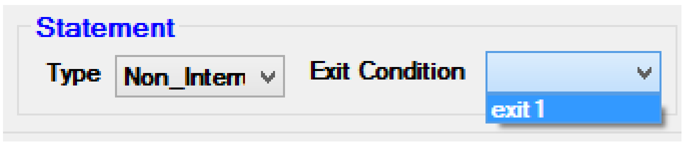


```
(US_Front>=10&&US_Front<=50)||(CS_Left>=10&&CS_Left<=30)
```

You can combine the use of advanced conditions and basic conditions by entering code into the advanced condition and also setting conditions using the slider. You can then set the advanced condition to 'and' or 'or' the two conditions:

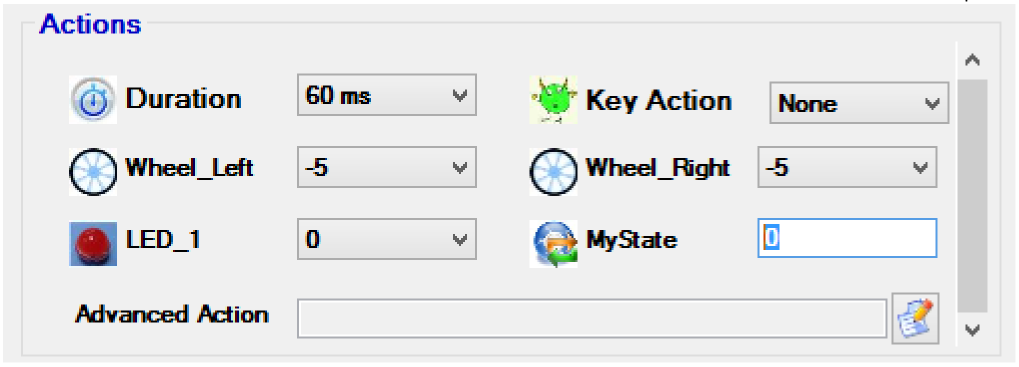


To now set the condition created as the exit save and exit the state manager and set the statement type of non-interrupt. The exit condition can now be selected the statement from the exit condition drop box menu:



7.7 Setting Actions

The action for a particular statement can be added using the actions box in the AI panel. This action will be run if the condition is true and it is the highest priority true statement. It will run once after which the condition is checked again and if it is still true it will execute again.

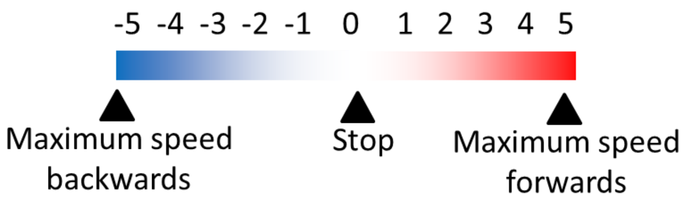


Duration: 60 ms
Key Action: None
Wheel_Left: -5
Wheel_Right: -5
LED_1: 0
MyState: 0

Duration is used to specify the duration for the action. The action will be continuously executed for the period that is specified in duration. The minimum unit for duration is 60ms.

7.7.1 Controlling the motors

The two wheels can be controller independently. Their direction and speed can be controlled. Each wheel can be set to a value between -5 and 5. Where +5 is maximum speed forward, -5 is maximum speed backwards and 0 is stop.



The duration for the motor movement can be set. The minimum amount of time that the wheel action can be set for is 60ms.

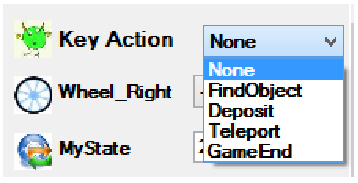
7.7.2 LEDs

The LEDs can be set by setting the LED value to different values:

LED Value	LED
0	LED off
1	LED blinks
2	LED is steady on

7.7.3 Key Actions

For certain tasks, a key action must be specified by choosing a the action from a drop down menu:



- **FindObject** – select this when an object is detected
- **Deposit** – when the deposition area is found
- **Teleport** – select this to teleport from world 1 to world 2
- **GameEnd** – select this to end the game

7.7.4 Setting 'MyState'

The ‘mystate’ value determines whether you are in World 1 or World 2. Setting this allows you to teleport from World 1 to World 2.

7.7.5 Advanced Actions

8 Changing the fields